Pop Middle

Mr. Panda wants to start this Practical Exam off with a simple task. He wants you to simulate a **Queue** that can support the following operations:

|  |  |
| --- | --- |
| **Operation** | **Description** |
| PUSH **[x]** | Push **[x]** into the queue |
| POP | Remove the **front** element of the queue and output its value. It is guaranteed there will be **at least one element in the queue** when this operation happens. |
| POPMIDDLE | Remove the **middle** element of the queue and output its value. It is guaranteed there will be an **odd number of elements in the queue** when this operation happens. |

Lastly, Mr. Panda wants you to list all the values in the queue from front to back. It is guaranteed there will be **at least one element in the** queue at the end of all the operations.

**Input**

The first line of input contains an integer **Q**. **Q** lines will follow, representing an operation each. The operations should be executed in order and the format would be as described in the table above. (See sample)

**Output**

For every **POP** and **POPMIDDLE** operation, output the value removed.

At the end of all **Q** operations, output all the values in the queue from front to back. Add a single space between two consecutive values. **Do not print a space after the last value.** Instead, remember to print an end-line character at the end of the output.

**Limits**

* 1 ≤ **Q** ≤ 400,000
* All the values will range from 1 to 109 inclusive.
* It is guaranteed there is **at least one element in the queue** when a **POP** operation happens
* It is guaranteed there is **an odd number of elements in the queue** when a **POPMIDDLE** operation happens.
* It is guaranteed there is **at least one element in the queue** at the end of all the operations

|  |  |
| --- | --- |
| Sample Input (**popmiddle1.in**) | Sample Output (**popmiddle1.out**) |
| 13  PUSH 1  POP  PUSH 2  POPMIDDLE  PUSH 5  PUSH 3  PUSH 4  PUSH 4  POP  PUSH 2  PUSH 2  POPMIDDLE  PUSH 4 | 1  2  5  4  3 4 2 2 4 |

**Explanation**

The first element 1 is pushed and then popped. The next element 2 is pushed and then popped again since it is the only element in the queue and thus also the middle element. The next 4 elements pushed will give [5, 3, 4, 4] in the queue. The next pop will remove the element 5 which is at the front, giving [3, 4, 4]. The next 2 pushes will give [3, 4, 4, 2, 2] in the queue and the middle element is 4 which is popped, leaving [3, 4, 2, 2]. The last push will give [3, 4, 2, 2, 4] in the queue.